

Specyfikacja wymagań systemowych

3@KASK

5 maja 2009

Symbol projektu: 3@KASK	Opiekun projektu: Tomasz Boiński
Nazwa Projektu: Wizualizacja grafów za pomocą biblioteki Prefuse	

Nazwa Dokumentu: Specyfikacja wymagań systemowych	Nr wersji: 0.0
Odpowiedzialny za dokument: Piotr Orłowski	Data pierwszego sporządzenia: 15 kwietnia 2009
Przeznaczenie: ???	Data ostatniej aktualizacji: 5 maja 2009

Historia dokumentu

Wersja	Opis modyfikacji	Rozdział/strona	Autor modyfikacji	Data
1	Stworzenie	wszystkie	Grupa projektowa	15.04.09
2	Wpisanie celów i wymogów ogólnych	cele	Grupa projektowa	16.04.09
3	Wpisanie funkcjonalności wizualizacyjnych		Grupa projektowa	28.04.09

Spis treści

1	Cele systemu	3
1.1	Cele biznesowe	3
1.2	Cele funkcjonalne	3
2	Otoczenie systemu	4
2.1	Użytkownicy	4
2.2	Systemy zewnętrzne	4
3	Przewidywane komponenty systemu	4
3.1	Podsystemy	4
3.2	Komponenty sprzętowe	4
3.3	Programowe	4
4	Wymagania funkcjonalne	5
4.1	Wymagania wizualizacji ontologii	5
4.2	Projekt wizualizacji	6
5	Wymagania na dane	8
6	Wymagania jakościowe	8
6.1	Wymagania w zakresie wiarygodności	8
6.2	Wymagania w zakresie wydajności	8
6.3	Wymagania w zakresie elastyczności	8
6.4	Wymagania w zakresie użyteczności	9
7	Sytuacje wyjątkowe	9
8	Dodatkowe wymagania	9
8.1	Wymagania sprzętowe	9
8.2	Wymagania programowe	9
8.3	Inne wymagania	10
9	Kryteria akceptacyjne	10
	Literatura	11

1 Cele systemu

1.1 Cele biznesowe

Cele biznesowe precyzują korzyści związane z wdrożeniem systemu.

CB001	Ułatwienie pracy programistom tworzącym aplikacje wizualizujące ontologie
Opis:	Istnieje zapotrzebowanie na bibliotekę tłumaczącą OWL bezpośrednio na elementy graficzne.
Źródło:	Wstępna specyfikacja projektu
Priorytet:	bardzo ważne
CB002	Ułatwienie zakończenia projektu OCS
Opis:	Moduł wizualizujący ontologię w OCS wymaga modernizacji i rozbudowy funkcjonalności. Zapewnienie biblioteki wizualizującej ontologię ułatwi i przyspieszy ten proces.
Źródło:	Klient - mgr inż. Tomasz Boiński
Priorytet:	bardzo ważne
CB003	Zwiększenie atrakcyjności portalu OCS
Opis:	Poprawa estetyki modułu wizualizującego ontologię może przyczynić się do sukcesu portalu po jego wdrożeniu.
Źródło:	Klient - mgr inż. Tomasz Boiński
Priorytet:	mało ważne
CB004	
Opis:	
Źródło:	
Priorytet:	

1.2 Cele funkcjonalne

Cele funkcjonalne wymieniają główne funkcje, które ma spełniać system.

CF001	Intuicyjne API
Opis:	
Źródło:	
Priorytet:	średnio ważne
CF002	Dobra dokumentacja
Opis:	Przygotowanie dokumentacji w Javadoc ułatwi pracę użytkownikom biblioteki.
Źródło:	Klient - mgr inż. Tomasz Boiński
Priorytet:	bardzo ważne
CF003	Wizualizacja ontologii
Opis:	Stworzenie biblioteki, która pozwoli na wizualizację obiektów OWL API przy użyciu odpowiedniej biblioteki graficznej.
Źródło:	Specyfikacja projektu
Priorytet:	bardzo ważne
CF004	Umożliwienie graficznej edycji i dodawania obiektów OWL API
Opis:	Dostarczenie tej funkcjonalności ułatwi tworzenie programów z interfejsem pozwalającym na edycję ontologii zapisanych w OWL API.
Źródło:	Klient - mgr inż. Tomasz Boiński
Priorytet:	średnio ważne

CF005	Udostępnienie informacji do debuggowania
Opis:	Biblioteka powinna wysyłać komunikaty informacyjne, ostrzegawcze oraz informujące o błędach na strumień udostępniony użytkownikowi.
Źródło:	Standard tworzenia biblioteki [1]
Priorytet:	średnio ważne

2 Otoczenie systemu

Zespół projektowy musi poznać otoczenie, w jakim ma pracować system. Z rozmów z klientem powinno dać się wyszczególnić użytkowników oraz systemy zewnętrzne. Jeśli się nie da, to otoczenie systemu trzeba będzie zdefiniować w trakcie analizy funkcjonalnej.

2.1 Użytkownicy

Specyfika projektu nie definiuje użytkowników systemu. Tutaj jest ID	A tutaj nazwa
Opis:	
Potrzeby:	
Zadania:	
Źródło:	
Priorytet:	

2.2 Systemy zewnętrzne

Specyfika systemu nie wymaga definiowania systemów zewnętrznych.

Tutaj jest ID	A tutaj nazwa
Opis:	
Interfejsy:	
Źródło:	
Priorytet:	

3 Przewidywane komponenty systemu

3.1 Podsystemy

Specyfika projektu sprawia, że podsystemy nie będą rozpatrywane.

3.2 Komponenty sprzętowe

Specyfika projektu sprawia, że komponenty sprzętowe nie będą rozpatrywane.

3.3 Programowe

Tutaj jest ID	Prefuse
Opis:	Biblioteka graficzna do wizualizacji grafów w języku Java
Powiązania:	Java
Źródło:	http://prefuse.org/
Priorytet:	bardzo ważne

4 Wymagania funkcjonalne

Wymagania funkcjonalne stanowią mocno rozbudowaną część specyfikacji. Można je podzielić na grupy dotyczące różnych zadań, różnych użytkowników (systemów zewnętrznych) albo różnych komponentów.




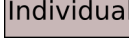

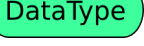


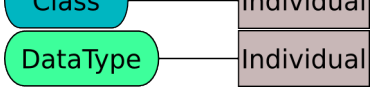
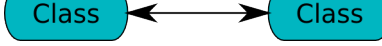
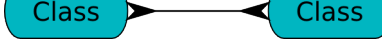
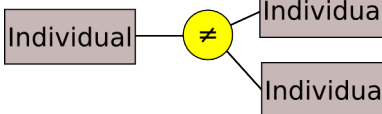
WF001	Udostępnienie kilku algorytmów wizualizacji
Opis:	Biblioteka powinna udostępniać kilka trybów prezentacji grafów (np. w formie drzewa, w formie gwiazdy i innych).
Dotyczy:	
Źródło:	klient - mgr Tomasz Boiński
Powiązania:	Wizualizacja ontologii (CF003)
Priorytet:	ważny
WF002	Parametryzacja trybów wizualizacyjnych
Opis:	Domyślne parametry w trybach wizualizacji (takie jak długość krawędzi grafu, automatyczne układanie) powinny zostać dobrane w taki sposób, by obraz był przejrzysty, stabilny i czytelny.
Dotyczy:	CF003
Źródło:	klient - mgr Tomasz Boiński
Powiązania:	
Priorytet:	średnio ważny
WF003	Udostępnienie strumienia błędów.
Opis:	Biblioteka będzie udostępniać strumień danych, w którym znajdują się komunikaty o błędach. Strumień ten będzie mógł zostać wykorzystany przez użytkownika.
Dotyczy:	CF005
Źródło:	klient - mgr inż. Tomasz Boiński
Powiązania:	
Priorytet:	ważne

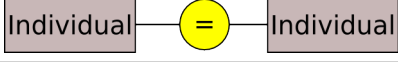
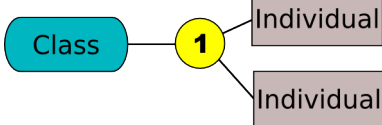
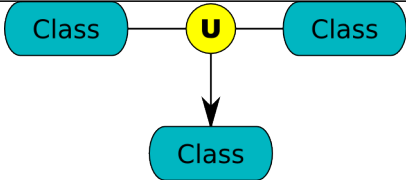
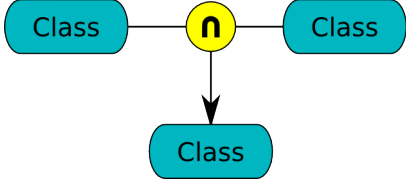
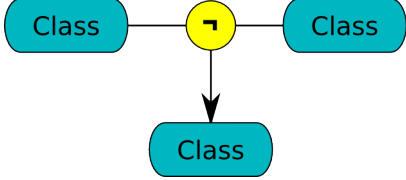
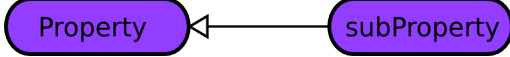






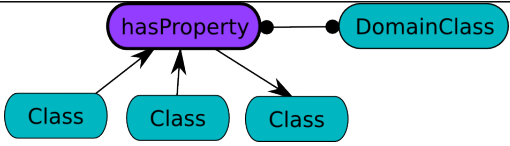
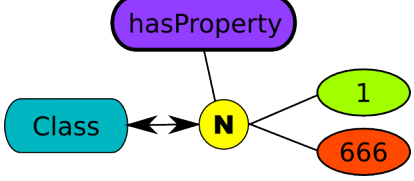
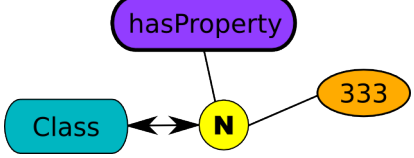
4.1 Wymagania wizualizacji ontologii

WF004	Rozróżnialność podstawowych symboli
Opis:	Class, Individual, Property powinny mieć rozróżnialne symbole
Dotyczy:	CF003
Źródło:	klient - mgr inż. Tomasz Boiński
Powiązania:	
Priorytet:	bardzo ważne
WF005	Rozróżnialność szczególnych typów Class
Opis:	Klasa anonimowa, datatype, Thing i Nothing powinny być łatwo rozpoznawalne.
Dotyczy:	CF003
Źródło:	klient - mgr inż. Tomasz Boiński
Powiązania:	WF004
Priorytet:	ważne
WF006	Rozróżnialność związków między klasami (Class), instancjami (Individual) oraz predykatami (Property)
Opis:	Różne symbole dla equivalentClass, disjointWith, subclassOf, sameAs, differentFrom, allDifferent, oneOf, unionOf, intersectionOf, complementOf, subProperty, equivalentProperty, hasProperty.
Dotyczy:	CF003
Źródło:	klient - mgr inż. Tomasz Boiński
Powiązania:	WF005, WF004
Priorytet:	ważne

WF007	Rozróżnialność ograniczeń predykatów (Restrictions).
Opis:	Wyróżnić kardynalność (cardinality), domeny (domains) predykatów, inverseOf, właściwości predykatów (transitive, symmetric, functional, inverseFunctional).
Dotyczy:	CF003
Źródło:	klient - mgr inż. Tomasz Boiński
Powiązania:	WF004
Priorytet:	ważne
WF008	Podświetlanie wybranych związków i powiązań.
Opis:	Podświetlać subklasy danej klasy po ich wybraniu myszką po zdefiniowanym zdarzeniu; podobnie subproperty i complex class.
Dotyczy:	CF003
Źródło:	klient - mgr inż. Tomasz Boiński
Powiązania:	WF006
Priorytet:	mało ważne
WF009	Możliwość definiowania zdarzeń.
Opis:	Użytkownik będzie mógł pod uchwyt zdarzeń podpinąć własne funkcje obsługi.
Dotyczy:	CF003, CF004
Źródło:	klient - mgr inż. Tomasz Boiński
Powiązania:	
Priorytet:	mało ważne

4.2 Projekt wizualizacji

Identyfikator:	Nazwa	Wizualizacja
PW001:	Thing	
PW002:	Nothing	
PW003:	Class	
PW004:	Individual	
PW005:	Property	
PW006:	Datatype	
PW007:	Anonymous Class	
PW008:	Subclass	
PW009:	instanceOf	
PW010:	equivalentClass	
PW011:	disjointWith	
PW012:	differentFrom / allDifferent	

PW013:	sameAs	
PW014:	oneOf	
PW015:	unionOf	
PW016:	intersectionOf	
PW017:	complementOf	
PW018:	subProperty	
PW019:	inverseOf (property)	
PW020:	equivalentProperty	
PW021:	functionalProperty	
PW022:	inverseFunctionalProperty	
PW023:	symmetricProperty	
PW024:	transitiveProperty	
PW025:	hasProperty / domain	
PW026:	minCardinality / maxCardinality	
PW027:	cardinality	

5 Wymagania na dane

Wymagania na dane pomagają w określeniu, jakie dane będą przetwarzane w systemie. Nie trzeba precyzować wszystkich danych. Szczegóły znajdują się w projekcie bazy danych.

WD001	Obsługa obiektów OWL API
Opis:	Bibliotek będzie przystosowana do pobierania, obróki i zwracania obiektów OWL API
Powiązania:	
Źródło:	Klient - mgr inż. Tomasz Boiński
Priorytet:	bardzo ważne

6 Wymagania jakościowe

Określenie wymagań jakościowych ułatwia późniejsze uzyskanie wysokiej jakości systemu. Podział wymagań jakościowych na kategorie jest związany z drzewem jakości (dotyczy wszystkich gałęzi drzewa za wyjątkiem funkcjonalności).

6.1 Wymagania w zakresie wiarygodności

Wymagania w zakresie wiarygodności będą rozszerzały wymagania funkcjonalne.

WJ001	Poprawność wizualizacji
Opis:	Wszystkie wizualizowane elementy powinny pochodzić z ontologii otrzymanej na wejściu programu. Program nie powinien dodawać własnych elementów (np. wywnioskowanych).
Powiązania:	WJ002
Źródło:	klient - mgr inż. Tomasz Boiński
Priorytet:	bardzo ważne
WJ002	Kompletność wizualizacji
Opis:	Jeżeli biblioteka nie wizualizuje danej funkcji OwlAPI informacja o tym powinna znaleźć się w strumieniu błędów.
Powiązania:	CF005, WJ001, WD001
Źródło:	klient - mgr inż. Tomasz Boiński
Priorytet:	ważne

6.2 Wymagania w zakresie wydajności

Wymagania w zakresie wydajności będą miały zastosowanie w czasie projektowania architektury systemu. Brak wymogów wydajnościowych ze względu na specyfikę projektu.

Tutaj jest ID	A tutaj nazwa
Opis:	
Powiązania:	
Źródło: klient - mgr inż. Tomasz Boiński	
Priorytet:	

6.3 Wymagania w zakresie elastyczności

Wymagania w zakresie elastyczności będą miały zastosowanie w czasie wyboru koncepcji systemu.

WJ003 ID	Obsługiwane wersje Javy.
Opis:	Biblioteka powinna wspierać wersje Javy 1.5 i nowsze.
Powiązania:	
Źródło:	klient - mgr inż. Tomasz Boiński
Priorytet:	bardzo ważne
WJ004 ID	Obsługiwane wersje OwlAPI.
Opis:	Powinna istnieć możliwość podpięcia zewnętrznego OwlAPI (wybranego przez użytkownika/programistę).
Powiązania:	
Źródło:	klient - mgr inż. Tomasz Boiński
Priorytet:	bardzo ważne

6.4 Wymagania w zakresie użyteczności

Wymagania w zakresie użyteczności będą brane pod uwagę głównie w czasie projektowania interfejsu użytkownika.

Ze względu na przyjętą metodykę wytwarzania oprogramowania zagadnienie to zostanie rozpatrzone w przyszłości.

Tutaj jest ID	A tutaj nazwa
Opis:	
Powiązania:	
Źródło:	
Priorytet:	

7 Sytuacje wyjątkowe

Sytuacje wyjątkowe stanowią dalsze rozszerzenie wymagań funkcjonalnych i wiarygodnościowych.

Ze względu na specyfikę projektu sytuacje wyjątkowe nie będą rozpatrywane.

Tutaj jest ID	A tutaj nazwa
Opis:	
Powiązania:	
Źródło:	
Priorytet:	

8 Dodatkowe wymagania

W tym miejscu podaje się te wymagania, które nie mieszczą się w zakresie poprzednich kategorii wymagań.

8.1 Wymagania sprzętowe

Wymagania sprzętowe można by umieścić w ramach specyfikacji komponentów sprzętowych, ale jeśli jest wiele komponentów sprzętowych różnych z punktu widzenia funkcjonalnego, ale o wspólnych wymaganiach sprzętowych, to można te wymagania umieścić właśnie tutaj.

Ze względu na specyfikę projektu wymagania sprzętowe nie będą rozpatrywane.

Tutaj jest ID	A tutaj nazwa
Opis:	
Dotyczy:	
Źródło:	
Priorytet:	

8.2 Wymagania programowe

Trzeba odróżniać rzeczywiste wymagania programowe klienta od jego sugestii (np. przez podanie opcjonalnego priorytetu).

WD003	JVM
Opis:	Do skorzystania z biblioteki niezbędna jest JVM.
Dotyczy:	CF001, CF002
Źródło:	klient - mgr inż. Tomasz Boiński
Priorytet:	ważne

8.3 Inne wymagania

WD001	Dokumentacja w javadoc
Opis:	Wszystkie ważne klasy i funkcje powinny mieć odpowiednią dokumentację w formacie javadoc.
Dotyczy:	CF001, CF002
Źródło:	klient - mgr inż. Tomasz Boiński
Priorytet:	ważne
WI002	Dokumentacja w języku angielskim
Opis:	Dokumentacja wszystkich funkcji i klas powinna posiadać angielską wersję językową.
Dotyczy:	CF001, CF002
Źródło:	klient - mgr inż. Tomasz Boiński
Priorytet:	mało ważne
WI003	Dokumentacja w języku polskim
Opis:	Dokumentacja wszystkich funkcji i klas powinna posiadać polską wersję językową.
Dotyczy:	CF001, CF002
Źródło:	klient - mgr inż. Tomasz Boiński
Priorytet:	ważne
WI004	Nazwy zmiennych i funkcji w języku angielskim
Opis:	Nazwy zmiennych i funkcji powinny zostać dobrane w języku angielskim i zgodnie ze standardami programowania w javie[standardJava].
Dotyczy:	CF001, CF002
Źródło:	klient - mgr inż. Tomasz Boiński
Priorytet:	ważne

9 Kryteria akceptacyjne

Tu podać kryteria, jakim zostanie poddany gotowy system przed ostatecznym jego przyjęciem.

KA001	Spełnione są podstawowe wymagania wymienione w dokumencie SWS.
Opis:	Spełnione są wszystkie wymagania ważne i bardzo ważne zdefiniowane w SWS.
Dotyczy: W*	
Źródło:	klient - mgr inż. Tomasz Boiński
Priorytet:	ważne

Literatura

- [1] Greg Travis. Build your own java library. publikacja elektroniczna.
[http://www.digilife.be/quickreferences/PT/Build your own Java library.pdf](http://www.digilife.be/quickreferences/PT/Build%20your%20own%20Java%20library.pdf).